

## Unit 3: working With Dart

### 3.1 DART overview, concept, features and installation

In today's era, we make use of Mobile and Internet for every day to day tasks like shopping, office meetings, online classes, communication, sharing, entertainment, gaming and so on. We find an App or a Website for everything we need. All business adopted the use of app and website for their business ease and marketing. Now a days all small, medium or large business use apps and websites to enlarge their business and to reach to their customers easily. Businesses now need apps and website for all their front business with customers to make it easy for their customers. Even government have taken an advantage of apps and websites to easily communicate with citizens and provide benefits of government schemes to all general people very effectively. During Corona lockdown, Apps, Internet and Websites play tremendous role to keep Education, Government Healthcare operations and Banking on track. So now if you are thinking for new business or want to launch your new idea in reality, first thing you need is Website and Mobile App.

## Introduction to Dart

The Dart language is a core of Flutter Framework. Modern framework like Flutter requires high language, capable of providing best experience to developers to create awesome mobile application.

Dart is an open-source general-purpose programming language. It is originally developed by Google and later approved as a standard by ECMA. Dart is a new programming language meant for the server as well as the browser. Introduced by Google, the Dart SDK ships with its compiler – the Dart VM. The SDK also includes a utility -dart2js, a transpiler that generates JavaScript equivalent of a Dart Script. This tutorial provides a basic level understanding of the Dart programming language.

Syntactically, Dart bears a strong resemblance to Java, C, and JavaScript. It is a dynamic object-oriented language with closure and lexical scope. The Dart language was released in 2011 but came into popularity after 2015 with Dart 2.0.

It is a compiled language and supports two types of compilation techniques.

- **AOT (Ahead of Time)** - It converts the Dart code in the optimized JavaScript code with the help of the dar2js compiler and runs on all modern web-browser. It compiles the code at build time.
- **JOT (Just-In-Time)** - It converts the byte code in the machine code (native code), but only code that is necessary.

## Benefit of using Dart

Dart aggregate the benefit of most of the high level language with mature language features as follows

1. **Productive tooling:** This include tools to analyze code, IDE plugins and big package ecosystem.
2. **Garbage collection:** Deals with memory deallocation
3. **Type annotation:** Maintain Security and consistency for all kind of data in application
4. **Static typed:** Dart is type-safe and uses type inference to analyze types in runtime. This feature is important for finding bugs during compile time.
5. **Portability:** This is not only for the web, but it can be natively compiled to ARM and x86 code.

## Features of Dart Language

The Dart is an object-oriented, open-source programming language which contains many useful features. It is the new programming language and supports an extensive range of programming utilities such as interface, collections, classes, dynamic and optional typing. It is developed for the server as well as the browser. Below is the list of the important Dart features.

### Open Source

Dart is an open-source programming language, which means it is freely available. It is developed by Google, approved by the ECMA standard, and comes with a BSD license.

### Platform Independent

Dart supports all primary operating systems such as Windows, Linux, Macintosh, etc. The Dart has its own Virtual Machine which known as Dart VM, that allows us to run the Dart code in every operating system.

### Object-Oriented

Dart is an object-oriented programming language and supports all oops concepts such as classes, inheritance, interfaces and optional typing features. It also supports advance concepts like mixin, abstract, classes, reified generic, and robust type system.

### **Concurrency**

Dart is an asynchronous programming language, which means it supports multithreading using Isolates. The isolates are the independent entities that are related to threads but don't share memory and establish the communication between the processes by the message passing. The message should be serialized to make effective communication. The serialization of the message is done by using a snapshot that is generated by the given object and then transmits to another isolate for de-serializing.

### **Extensive Libraries**

Dart consists of many useful inbuilt libraries including SDK (Software Development Kit), core, math, async, http, convert, html, IO, etc. It also provides the facility to organize the Dart code into libraries with proper namespacing. It can reuse by the import statement.

### **Easy to learn**

As we discussed in the previous section, learning the Dart is not the Hercules task as we know that Dart's syntax is similar to Java, C#, JavaScript, kotlin, etc. if you know any of these languages then you can learn easily the Dart.

### **Flexible Compilation**

Dart provides the flexibility to compile the code and fast as well. It supports two types of compilation processes, AOT (Ahead of Time) and JIT (Just-in-Time). The Dart code is transmitted in the other language that can run in the modern web-browsers.

### **Type Safe**

The Dart is the type safe language, which means it uses both static type checking and runtime checks to confirm that a variable's value always matches the variable's static type, sometimes it known as the sound typing.

## **Dart Installation**

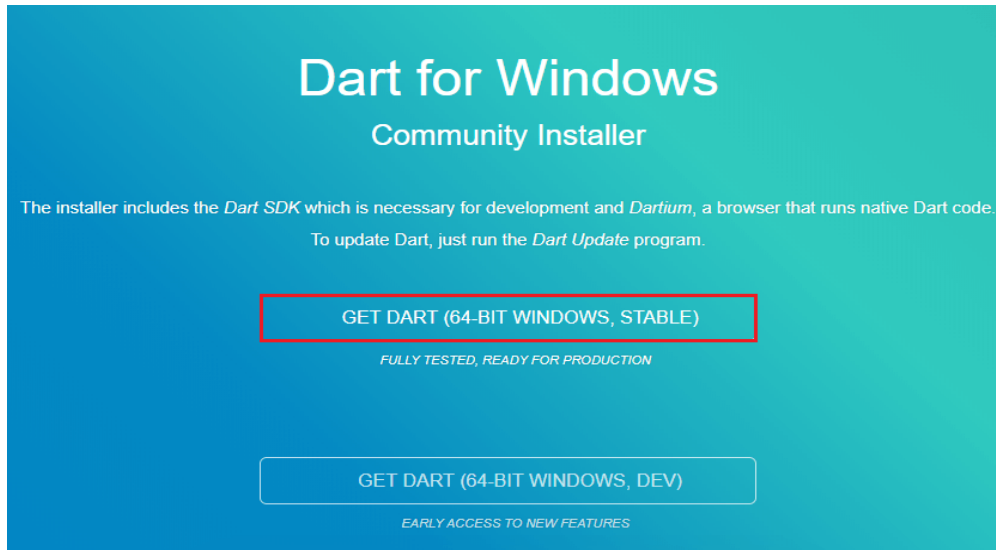
### **Install Dart on Windows**

Follow the below instructions to install Dart SDK in **Windows**.

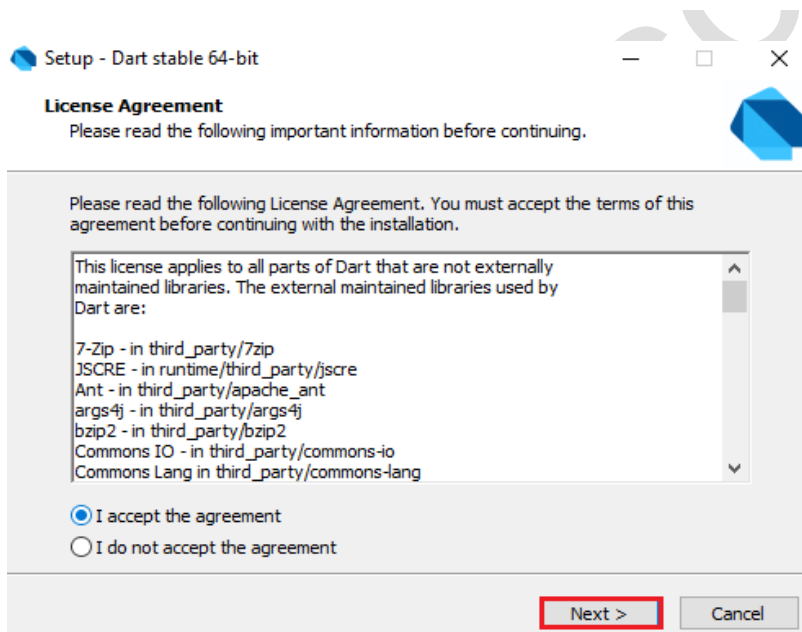
**Step -1:** Go to the browser and type the following link to download the SDK.

<http://www.gekorm.com/dart-windows/>

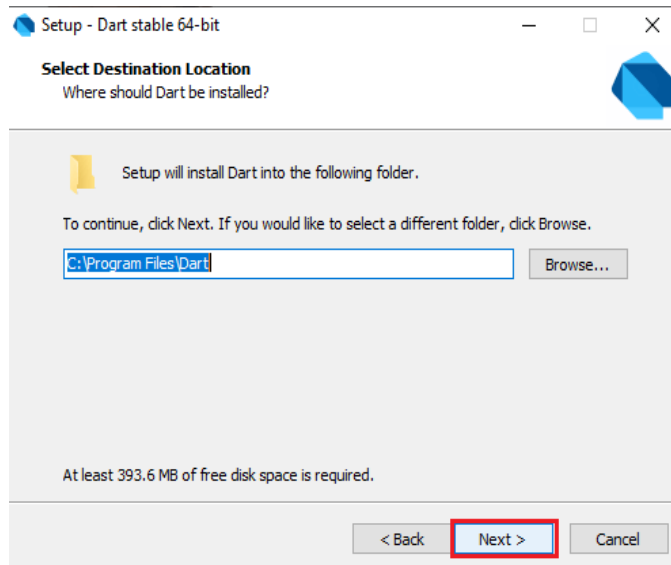
It will open the given page. Click on the following link.



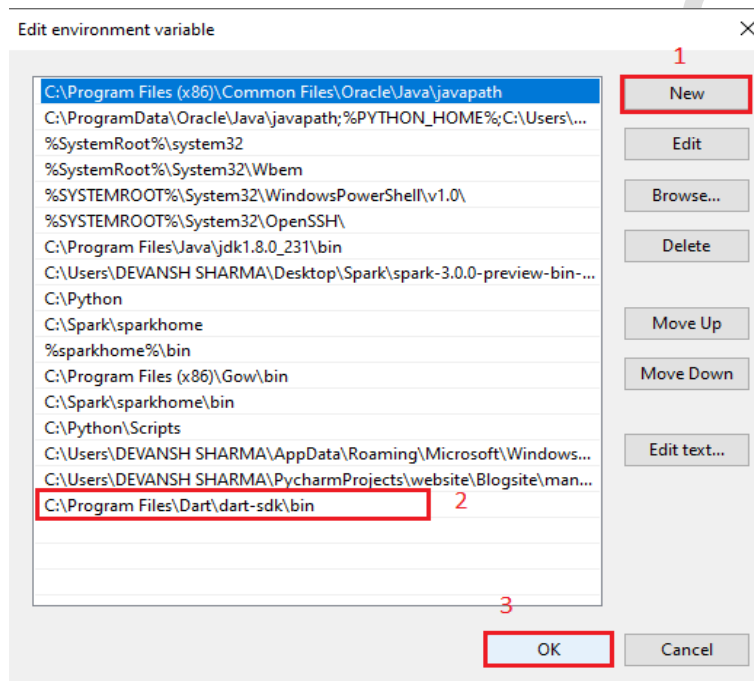
**Step - 2:** Run the Dart installer(It is the .exe file that we downloaded in the previous step) and click on the **Next** button.



**Step - 3:** It provides the option to select the Dart installation path. After the path is selected, click on the **Next** button.



**Step - 4:** After the download is completed, set the PATH environment variable to "C:\Program Files\Dart\dart-sdk\bin" in advance system properties.



**Step - 5:** Now open the terminal and verify the Dart installation by typing dart.

```

C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.18362.778]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\DEVANSH_SHARMA>dart
Usage: dart [<vm-flags>] <dart-script-file> [<script-arguments>]

Executes the Dart script <dart-script-file> with the given list of <script-arguments>.

Common VM flags:
--enable-asserts
  Enable assert statements.
--help or -h
  Display this message (add -v or --verbose for information about
  all VM options).
--package-root=<path> or -p<path>
  Where to find packages, that is, "package:..." imports.
--packages=<path>
  Where to find a package spec file.
--observe[=<port>[/<bind-address>]]
  The observe flag is a convenience flag used to run a program with a
  set of options which are often useful for debugging under Observatory.
  These options are currently:
    --enable-vm-service[=<port>[/<bind-address>]]
    --pause-isolates-on-exit
    --pause-isolates-on-unhandled-exceptions
    --warn-on-pause-with-no-debugger
  This set is subject to change.
  Please see these options (--help --verbose) for further documentation.
--write-service-info=<file_name>
  Outputs information necessary to connect to the VM service to the
  specified file in JSON format. Useful for clients which are unable to
  listen to stdout for the Observatory listening message.
--snapshot-kind=<snapshot_kind>
--snapshot=<file_name>
  These snapshot options are used to generate a snapshot of the loaded
  Dart script:
    <snapshot-kind> controls the kind of snapshot, it could be
      kernel(default) or app-jit
    <file_name> specifies the file into which the snapshot is written
--version
  Print the VM version.

C:\Users\DEVANSH_SHARMA>

```

If it is successfully installed then it looks like the above image.

## Install the Dart SDK on Linux

The steps of Dart installation on **Linux** is given below.

Before installing the **Dart**, if you are Debian/Ubuntu on AMD64(64-bit Intel) in your local machine, you can install the Dart through one of the following options.

- Install using apt-get
- Install a Debian package

### Installation using apt-get

**Step -1:** Type the following commands for a one-time setup.

1. \$sudo apt-get update
2. \$ sudo apt-get install apt-transport-https
- 3.

4. `$ sudo sh -c 'wget -qO- https://dl-ssl.google.com/linux/linux_signing_key.pub | apt-key add -'`
- 5.
6. `$ sudo sh -c 'wget -qO- https://storage.googleapis.com/download.dartlang.org/linux/debian/dart_stable.list > /etc/apt/sources.list.d/dart_stable.list'`

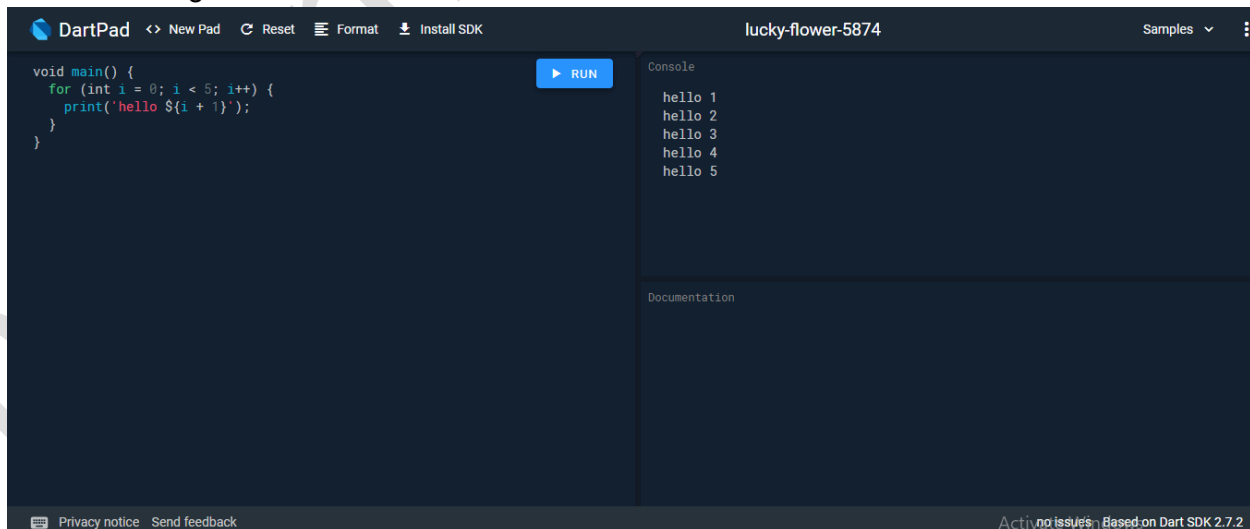
**Step - 2:** Type the following command in terminal to install the Dart SDK using apt-get option.

1. `$sudo apt-get update`
2. `$ sudo apt-get install dart`

It will successfully download the Dart SDK.

## 3.2 Online Editor Dartpad and Dart2js Tool

We have discussed Dart installation on the various operating systems so far, but if we do not want to install Dart then there is an online Dart editor (Known as DartPad) is available to run the Dart programs. The online DartPad is provided at <https://dartpad.dev/>. The DartPad offers to execute the dart scripts and display HTML and also console output. The online DartPad looks like the below image.



## The dart2js Tool

The Dart SDK comes with the dart2js tool, which transmits the Dart code into runnable JavaScript code. It is necessary because few web browsers do not support the Dart VM.

Use the following command in the terminal to compile the Dart code into JavaScript code.

1. `dart2js --out = <output_file>.js <dart_script>.dart`

The above command will create a file that contains the JavaScript code corresponding to the Dart code.

## Dart IDE Support

The Eclipse, IntelliJ, Android Studio and WebStorm are the IDEs from the Jet brains that support the Dart Programming, but WebStorm is more popular than others. We can download it from

<https://www.jetbrains.com/webstorm/download/#section=windows-version>.

## 3.3 Executing Dart basic code using Command line, DartPad and IDE

There are several ways to run the first program, which is given below:

- Using Command Line
- Running on Browser
- Using IDE

### Using Command Line

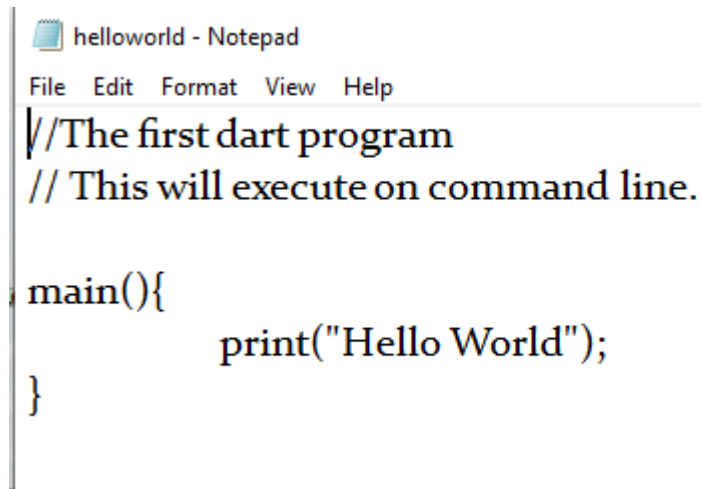
Step - 1:



Type **dart** on the terminal if it is showing dart runtime then Dart is successfully installed.

### Step - 2:

Open a text editor and create a file called "helloworld.dart". The file extension should be **.dart** that specifies; it is a Dart program file.



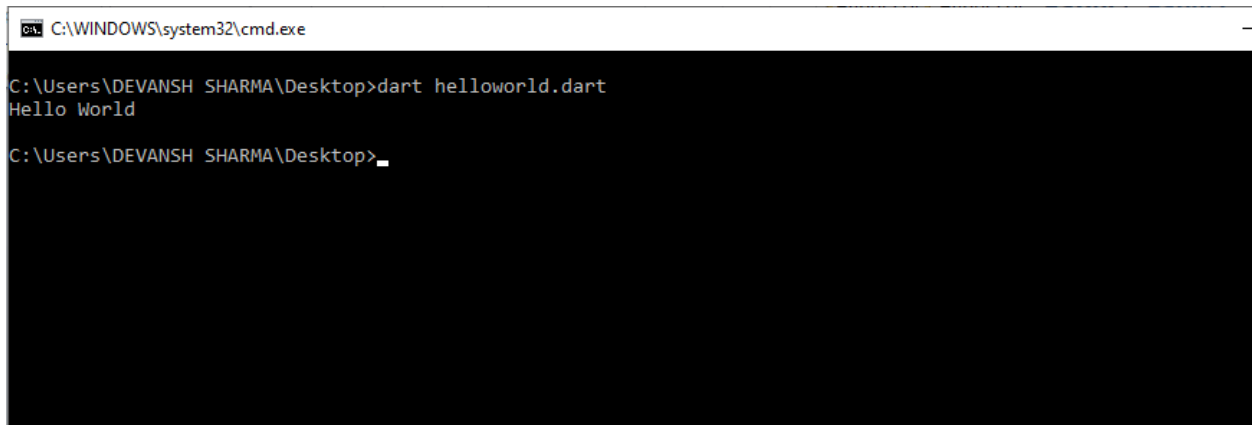
```
helloworld - Notepad
File Edit Format View Help
//The first dart program
// This will execute on command line.

main(){
    print("Hello World");
}
```

- **main()** - The **main()** function indicates that it is the beginning of our program. It is an essential function where the program starts its execution.
- **print()** - This function is used to display the output on the console. It is similar to C, JavaScript, or any other language. The curly brackets and semicolons are necessary to use appropriately.

### Step - 3:

Open the command line; compile the program run the Dart program by typing **dart helloworld.dart**. It will show **Hello World** on the screen.



```
C:\WINDOWS\system32\cmd.exe
C:\Users\DEVANSH SHARMA\Desktop>dart helloworld.dart
Hello World
C:\Users\DEVANSH SHARMA\Desktop>
```

## 3.4 Understanding Dart Syntax

### Dart Identifiers

Identifiers are the name which is used to define variables, methods, class, and function, etc. An Identifier is a sequence of the letters([A to Z],[a to z]), digits([0-9]) and underscore(\_), but remember that the first character should not be a numeric. There are a few rules to define identifiers which are given below.

- The first character should not be a digit.
- Special characters are not allowed except underscore (\_) or a dollar sign (\$).
- Two successive underscores (\_\_) are not allowed.
- The first character must be alphabet(uppercase or lowercase) or underscore.
- Identifiers must be unique and cannot contain whitespace.
- They are case sensitive. The variable name **Joseph** and **joseph** will be treated differently.

### Dart Printing and String Interpolation

The **print()** function is used to print output on the console, and **\$expression** is used for the string interpolation. Below is an example.

#### Example -

```
void main()
{
    var name = "Samir";
    var roll_no = 313;

    print("My name is ${name} My roll number is ${roll_no}");
}
```

#### Output:

My name is samir My roll number is 313

## Semicolon in Dart

The semicolon is used to terminate the statement that means, it indicates the statement is ended here. It is mandatory that each statement should be terminated with a semicolon(;).

# Dart Data Types

Dart supports the following built-in Data types.

- Number
- Strings
- Boolean
- Lists
- Maps
- Runes
- Symbols

## Dart Number

The Darts Number is used to store the numeric values. The number can be two types - integer and double.

- **Integer** - Integer values represent the whole number or non-fractional values. An integer data type represents the 64-bit non-decimal numbers between  $-2^{63}$  to  $2^{63}$ . A variable can store an unsigned or signed integer value. The example is given below -

```
int marks = 80;
```

- **Double** - Double value represents the 64-bit of information (double-precision) for floating number or number with the large decimal points. The double keyword is used to declare the double type variable.

```
double pi = 3.14;
```

## Dart Strings

A string is the sequence of the character. If we store the data like - name, address, special character, etc. It is signified by using either single quotes or double quotes. A Dart string is a sequence of UTF-16 code units.

```
var msg = "Welcome to JavaTpoint";
```

## Dart Boolean

The Boolean type represents the two values - true and false. The bool keyword uses to denote Boolean Type. The numeric values 1 and 0 cannot be used to represent the true or false value.

```
bool isValid = true;
```

## Dart Lists

In Dart, The list is a collection of the ordered objects (value). The concept of list is similar to an array. An array is defined as a collection of the multiple elements in a single variable. The elements in the list are separated by the comma enclosed in the square bracket[]. The sample list is given below.

```
var list = [1,2,3]
```

## Dart Maps

The maps type is used to store values in key-value pairs. Each key is associated with its value. The key and value can be any type. In Map, the key must be unique, but a value can occur multiple times. The Map is defined by using curly braces ({}), and comma separates each pair.

```
var student = {'name': 'Joseph', 'age':25, 'Branch': 'Computer Science'}
```

## Dart Dynamic Type

Dart is an optionally typed language. If the variable type is not specified explicitly, then the variable type is dynamic. The dynamic keyword is used for type annotation explicitly.

## Dart Variable

Variable is used to store the value and refer the memory location in computer memory. When we create a variable, the Dart compiler allocates some space in memory. The size of the memory block of memory is depended upon the type of variable. To create a variable, we should follow certain rules. Here is an example of a creating variable and assigning value to it.

## Rule to Create Variable

Creating a variable with a proper name is an essential task in any programming language. The Dart has some rules to define a variable. These rules are given below.

- The variable cannot contain special characters such as whitespace, mathematical symbol, runes, Unicode character, and keywords.
- The first character of the variable should be an alphabet([A to Z],[a to z]). Digits are not allowed as the first character.
- Variables are case sensitive. For example, - variable age and AGE are treated differently.
- The special character such as #, @, ^, &, \* are not allowed expect the underscore(\_) and the dollar sign(\$).
- The variable name should be retable to the program and readable.

### Syntax -

```
var <variable_name> = <value>;
```

```
var <variable_name>;
```

### Example -

```
var name = 'Samir'
```

## Default Value

While declaring the variable without initializing the value then the Dart compiler provides default value (Null) to the variable. Even the numeric type variables are initially assigned with the null value. Let's consider the following example.

```
int count;  
  
assert(count == null);
```

## Final and const

When we do not want to change a variable in the future then we use final and const. It can be used in place of **var** or in addition to a type. A final variable can be set only one time where the variable is a compile-time constant. The example of creating a final variable is given below.

### Example -

```
final name = 'Ricky';           // final variable without type annotation.  
  
final String msg = 'How are you?'; // final variable with type annotation.
```

If we try to change these values then it will throw an error.

```
name = 'Roger';                 // Error: Final variable can't be changed.
```

The **const** is used to create compile-time constants. We can declare a value to compile-time constant such as number, string literal, a const variable, etc.

```
const a = 1000;
```

The const keyword is also used to create a constant value that cannot be changed after its creation.

```
var f = const[];
```

If we try to change it, then it will throw an error.

```
f = [12]; //Error, The const variable cannot be change
```

## Comments

Dart provides three kinds of comments

### Single-line Comments

```
// This will print the given statement on screen
```

### Multi-line Comments

```
/* This is the example of multi-line comment  
This will print the given statement on screen */
```

### Documentation Comments

```
///This  
///is  
///a example of  
/// multiline comment
```